REG10J0119-0100

Renesas Starter Kit

for SH7086 Tutorial Manual

RENESAS SINGLE-CHIP MICROCOMPUTER

Rev.1.00
Revision date：17.Jan.2008

Renesas Technology Europe Ltd.
www.renesas.com

Table of Contents

Chapter 1. Preface 3
Chapter 2. Introduction 4
Chapter 3. Tutorial Project Workspace 5
Chapter 4. Project Workspace 6
4.1. Introduction 6
4.2. Creating a new Project Workspace 6
4.3. Build Configurations and Debug Sessions 7
4.3.1. Build Configuration 7
4.3.2. Debug Session 7
Chapter 5. Building the Tutorial Project 8
5.1. Building Code 8
5.2. Connecting the debugger 9
Chapter 6. Downloading and Running the Tutorial 10
Chapter 7. Project Files 14
7.1. Standard Project Files 14
7.1.1. Initialisation code (Resetprg.c / resetprg.h) 14
7.1.2. Board initialisation code (Hwsetup.c / hwsetup.h) 15
7.1.3. Main tutorial code (Main.c / main.h) 15
Chapter 8. Additional Information 16

Chapter 1. Preface

Cautions

This document may be, wholly or partially, subject to change without notice.

All rights reserved. No one is permitted to reproduce or duplicate, in any form, a part or this entire document without the written permission of Renesas Technology Europe Limited.

Trademarks

All brand or product names used in this manual are trademarks or registered trademarks of their respective companies or organisations.

Copyright

© Renesas Technology Europe Ltd. 2008. All rights reserved.

© Renesas Technology Corporation. 2008. All rights reserved.

© Renesas Solutions Corporation. 2008. All rights reserved.

Website: http://www.renesas.com/

Glossary

ADC	Analog to Digital Converter	LCD	Liquid Crystal Display
RSK	Renesas Starter Kit	USB	Universal Serial Bus
HEW	High performance Embedded Workshop	PC	Program Counter
CPU	Central Processing Unit	LED	Light Emitting Diode
E10A	'E10A for Starter Kit' debugger		

Chapter 2. Introduction

This manual is designed to answer, in tutorial form, the most common questions asked about using a Renesas Starter Kit (RSK): The tutorials help explain the following:

- How do I compile, link, download, and run a simple program on the RSK?
- How do I build an embedded application?
- How do I use Renesas' tools?

The project generator will create a tutorial project with two selectable build configurations

- 'Debug' is a project built with the debugger support included.
- 'Release' build demonstrating code suitable for release in a product.

Files referred to in this manual are installed using the project generator as you work through the tutorials. The tutorial examples in this manual assume that installation procedures described in the RSK Quick Start Guide have been completed. Please refer to the Quick Start Guide for details of preparing the configuration.

NOTE: These tutorials are designed to show you how to use the RSK and are not intended as a comprehensive introduction to the High performance Embedded Workshop (HEW) debugger or the compiler toolchains – please consult the relevant user manuals for more in-depth information.

Chapter 3. Tutorial Project Workspace

The workspace includes all of the files for two build configurations. The tutorial code is common to the “Debug” and “Release” build configurations. The tutorial is designed to show how code can be written, debugged then downloaded without the debug monitor in a ‘Release’ situation.

The build configuration menu in High-performance Embedded Workshop (HEW) allows the project to be configured such that certain files may be excluded from each of the build configurations. This allows the inclusion of the debug monitor within the Debug build, and its exclusion in the Release build. Contents of common C files are controlled with defines set up in the build configuration options and `#ifdef` statements within the source files.

Maintaining only one set of project files means that projects are more controllable.

Chapter 4.Project Workspace

4.1. Introduction

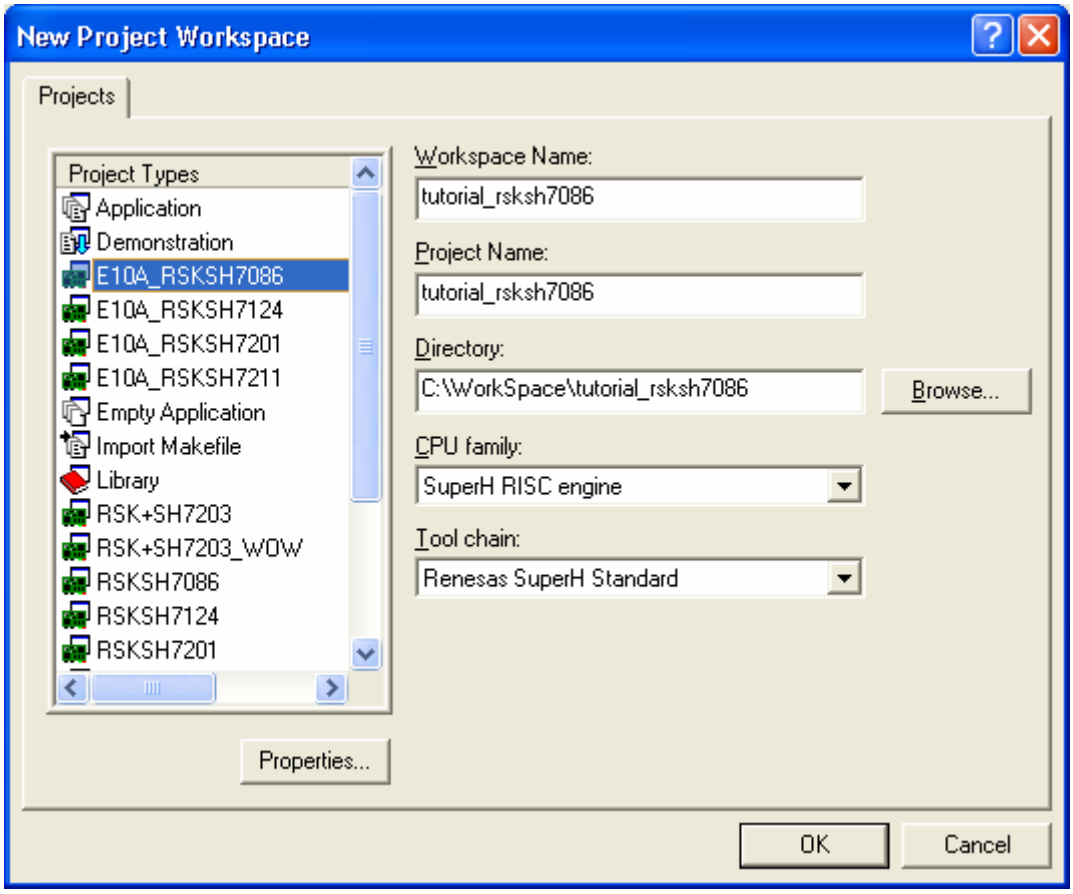
HEW is an integrated development tool that allows the user to write, compile, program and debug a software project on any of the Renesas Microcontrollers. HEW will have been installed during the software installation for the RSK product.

To begin using the RSK, this manual will describe the stages required to create and debug the supplied tutorial code.

4.2. Creating a new Project Workspace

To look at the program, start High-performance Embedded Workshop from the Windows Start Menu or from its icon:

Open a new tutorial workspace from the [File -> New Workspace…] menu or select 'Create a new project workspace' when presented with the 'Welcome!' dialog.



The example above shows the New Project Workspace dialog with the 'E10A_RSKSH7086' selected.

- Select the 'SuperH RISC Engine' CPU family and 'Renesas SuperH Standard' Toolchain for the RSK
- Select the 'E10A_RSKSH7086' Project type for the RSK from the project list.
- Enter a name for the workspace; all your files will be stored under a directory with this name.

- The project name field will be pre-filled to match the workspace name above; this name may be changed.
  Note: HEW allows you to add multiple projects to a workspace. You may add the sample code projects later so you may wish to choose a suitable name for the Tutorial project now.
- Click OK to start the RSK Project Generator wizard.

The next dialog presents the example projects available. Choose the Tutorial code which will be explained later in this manual. There is also an option for Sample code which provides examples for using various peripherals. This will open a new dialog allowing the selection of many code examples for the peripheral modules of the device. The final option is for an application build where the debugger is configured but there is no program code. This project is suitable for the user to add code without having to configure the debugger.

- Select 'Tutorial' as the type of project to generate and then click <Next>.
- Click <Finish> to create the project

The project generator wizard will display a confirmation dialog. Press <OK> to create the project and insert the necessary files.

A tree showing all the files in this project will appear in the HEW Workspace window.

- To view the file main.c, double-click on the file in the Workspace window. A new window will open showing the code.

4.3. Build Configurations and Debug Sessions

The workspace that has been created contains two Build Configurations and two Debug Sessions. The Build Configuration allows the same project to be built but with different compiler options. The options available to the user are described fully in the HEW Users Manual.

4.3.1. Build Configuration

The build configurations are selected from the left hand drop down list on the tool bar. The options available are "Debug" and "Release". The "Debug" builds are configured for use with the E10A debugger. The "Release" build is configured for final ROM-able code.

A common difference between the builds may be the optimisation settings. With Optimisation turned on the Debugger may seem to execute code in an unexpected order. To assist in debugging it is often helpful to turn off optimisation on the code being debugged.

- Select the 'Debug' build configuration in the left hand drop down list on the tool bar.

4.3.2. Debug Session

The debug sessions are selected from the right hand drop down list on the tool bar. The options vary between RSK however one will always start Debug and include the type of debug interface. The purpose of the debug sessions is to allow the use of different debugger targets or different debugger settings on the same project.

- Select the "SessionSH2_7086_E10A_USB_SYSTEM" session in the right hand drop down list on the tool bar.

Chapter 5. Building the Tutorial Project

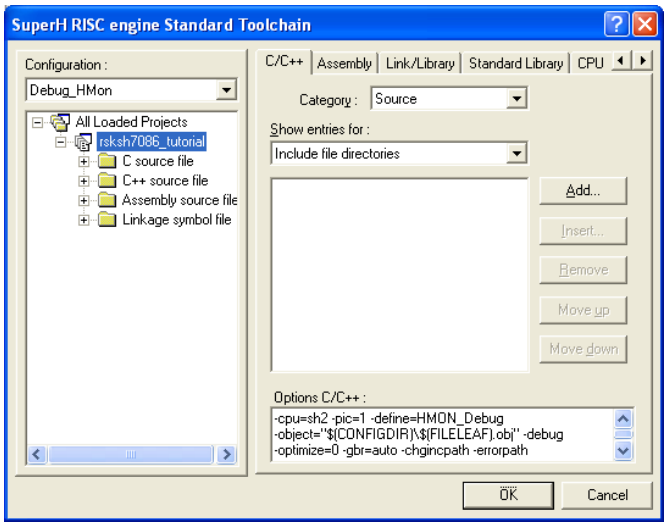
The tutorial project build settings have been pre-configured in the toolchain options. To view the tool chain options select the 'Build' Menu item and the relevant toolchain. This should be the first option(s) on the drop down menu.

The dialog that is displayed will be specific to the selected toolchain.

The configuration pane on the left hand side will exist on all the toolchain options. It is important when changing any setting to be aware of the current configuration that is being modified. If you wish to modify multiple or all build configurations this is possible by selecting 'All' or 'Multiple' from the 'Configuration' drop down list.

- Review the options on each of the tabs and 'Category' drop down lists to be aware of the options available.

When complete close the dialog box by clicking <OK>.



5.1. Building Code

There are three short cuts available for building the project.

- Select the 'Build All' tool bar button.

  This will build everything in the project that has not been excluded from the build. This includes the standard library.

- Select the 'Build' tool bar button.

  This will build all files that have changed since the last build. The standard library will not be built unless an option has been changed.

- Press 'F7'

  This is equivalent to pressing the 'Build' button described above.

- Build the project now by pressing 'F7' or pressing one of the build icons as shown above.

During the build each stage will be reported in the Output Window.

The build will complete with an indication of errors and warnings encountered during the build.

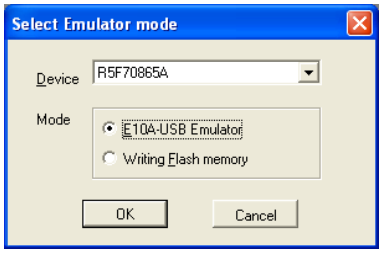
5.2. Connecting the debugger

For this tutorial it is necessary for you to power the RSK from the supplied PSU. The E10A module will be powered via the USB cable. Please be aware that if you have too many devices connected to your USB port it may be shut down by Windows. If this happens remove some devices and try again. Alternatively you can provide an external power source, taking care to ensure the correct polarity and voltage.

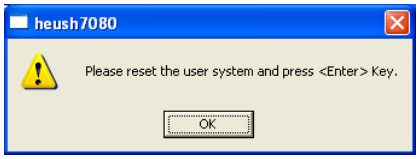
The Quick Start Guide provided with the RSK board gives detailed instructions on how to connect the E10A module to the host computer. The following assumes that the steps in the Quick Start Guide have been followed and the E10A module drivers have been installed.

- Connect the E10A module to the target hardware ensuring that it is plugged into the connector marked 'E10A'.
- Turn on the power to the RSK
- Connect the E10A to the host machine's USB port
- Click the <Connect> button on the debug toolbar.

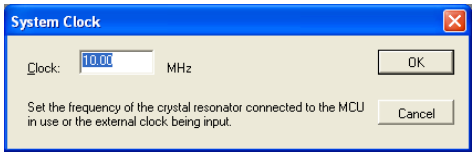
- The 'Select Emulator mode' dialog will appear. Select the device "R5F70865A" and the "E10A-USB Emulator" radio and click <OK>.



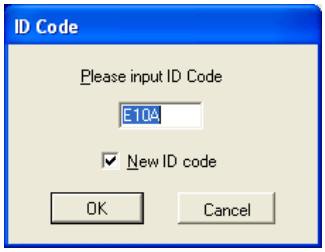
- A dialog will be shown asking you to reset the user system and press <Enter>, please do so.



- The 'System Clock' dialog will appear, enter "10" into the MHz box and click <OK>.



- The 'ID code' dialog will appear, enter "E10A" for the ID code and check the "New ID code" tick-box and click <OK>.
- HEW will connect to the target and show "Connected" in the 'Output' view.

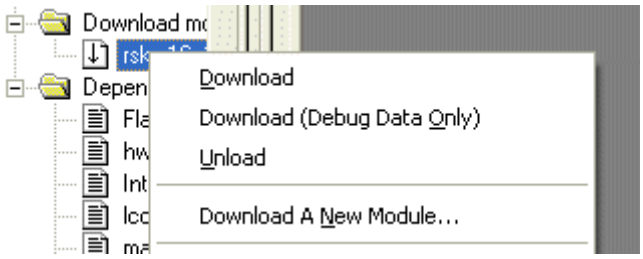


Chapter 6. Downloading and Running the Tutorial

Once the code has been built in HEW it needs to be downloaded to the RSK.

Now that you are connected to the target you should see an additional category in the workspace view called 'Download Modules'

- Right click on the download module listed and select 'Download'



On completion the debugger and code are ready to be executed.

To start debugging we need to reset the debugger and target.

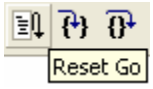
- Press 'Reset CPU' on the Debug Tool Bar.

The File window will open the Tutorial code at the entry point. An arrow marks the current position of the program counter.

```
00000800  void Initialise(void)
          {
              /* Initialize the Vector Base Register with the base address of exception
                 vector table. */
00000802      set_vbr((void *)((char *)0));

              /*  Copy initilized data from ROM to RAM.
                  dbsct.c file will be used by this function   */
00000806      _INITSCT();

              /* Set up the hardware  */
0000080C      HardwareSetup();

              /* Unmask all interrupts */
00000812      set_cr(SR_Init);

              /* Delay for one CPU cycle. */
00000816      nop();
```

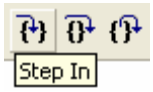
We will now skip over the initialisation code and proceed to the main tutorial.

- Open the file called 'resetprg.c' by double clicking it in the project navigator.
- Place a breakpoint at the call to main(); by double clicking in the column containing the PC arrow, next to the line to break at; or selecting the line and pressing F9; or right click on the line and select 'Toggle breakpoint'
- Press 'Reset Go' on the Debug Tool Bar.



The code will execute to the breakpoint. At this point all the device initialisation will have been completed.

- Press ‘Step In’ on the Debug Tool Bar.



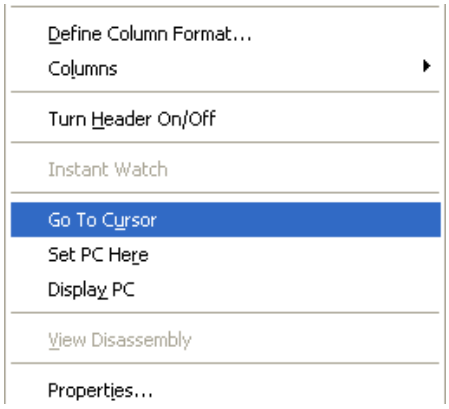
The code window will open ‘main.c’ and show the new position of the program counter.

```
00001624        void main(void)
                {
                    /*  Reset the LCD module. */
00001626            InitialiseDisplay();

                    /*  Display Renesas Splash Screen. */
0000162C            DisplayString(LCD_LINE1,"Renesas");
00001636            DisplayString(LCD_LINE2,NICKNAME);

                    /*  Flash the user LEDs for some time or until a key is pressed. */
00001640            FlashLEDs();

                    /*  Flash the user LEDs at a rate set by the user potentiomenter (ADC) us
                        interrupts. */
00001646            TimerADC();

                    /*  Demonstration of initialised variables.
                        Use this function with the debugger. */
0000164C            Statics_Test();
```

Support for the LCD display is included in the tutorial code. We do not need to be concerned about the details of the LCD interface – except that the interface is write-only and so is not affected if the LCD display is attached or not.

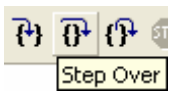
- Insert a breakpoint on the ‘TimerADC();’ function call.

- Right click on the ‘FlashLEDs();’ function and select ‘Go to cursor’.



The code will run to the selected line and stop. A temporary breakpoint was automatically inserted in the code and then removed when the program stopped at the breakpoint.

- Press ‘Step Over’ on the Debug Tool Bar.



The code will run and flash the LEDs 200 times. The debugger will not stop running until all 200 flashes have completed or a button is pressed on the RSK.

- If the LEDs are still flashing press the SW1 button on the RSK to exit the FlashLEDs() function.

The code will run to the breakpoint we previously set on the Timer function.

There are several versions of the timer function depending upon the peripherals available in the device. The default function is TimerADC which we shall demonstrate here.

The timer function initialises an interrupt on an available internal timer. On a compare match in the timer module an interrupt is generated. In the TimerADC code version the interrupt reads the last ADC conversion for the external potentiometer and uses the result to set the next compare match value. The ADC conversion is then re-started.

The interrupt initialisation is performed as part of the hardware setup. This is located in the file 'interrupts.c'.

- Open the file 'interrupts.c' by double clicking on the file in the workspace view.
- Review this file and find the interrupt function that changes the LED pins, INT_MTU1_TGIA1(void)
- Set a breakpoint on the line where the LED pins are modified.
- Press 'Go' or 'F5' to run the code from the current PC position.

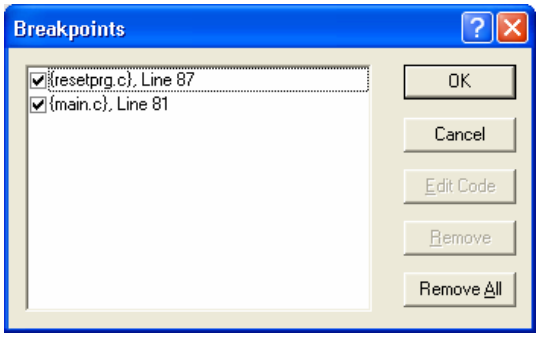
The code will stop in the interrupt routine. It is now possible to step through the interrupt function.

- Remove the breakpoint in the interrupt by double clicking again before exiting the function.
- Press 'Step Over' to step over the instruction and observe the LEDs turn off.
- Press 'Go' to run the code from the current PC position.

The code will now run to the infinite loop at the end of Main(). The user LEDs should now be flashing. If the RSK supports an ADC you can modify the flashing rate by adjusting the potentiometer on the board.

- Press 'Stop' on the debug tool bar.

- Press 'CTRL-B' to open the breakpoint window.
- Select 'Remove All'
- Press <OK>.

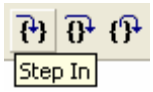


- Open the file 'main.c'
- Insert a breakpoint on 'StaticsTest();'.

The statics test is used to demonstrate that the initialisation has successfully copied all initialised variables from storage in flash to RAM.

- Press 'Reset Go' on the Debug Tool Bar.

The code will stop at the breakpoint. (Press a button to bypass the flashing LED test.)

- Press ‘Step In’ on the Debug Tool Bar. 

It is possible to monitor variables during debugging of the code. To set up a ‘watch’ on a variable place the mouse over the variable. If the variable is available in the current context a tool-tip will be displayed with the current value of the variable.

- Hover the mouse over the ‘ucStr’ variable to see the tooltip value. Then Right click on the variable name and select ‘Instant Watch’.

A dialog will open showing the variable and allowing further details to be explored.

- Press ‘Add’

The dialog will close and a new pane will open in the workspace containing the variable.

It is possible to see that the string has been successfully initialised to ‘ STATIC ‘.

- Set a breakpoint on the ‘DisplayString();’ function call inside the loop.
- Press ‘Go’ to run the code from the current PC position.

When the program stops you can see the modified string displayed on the second line of the LCD.

Inspection of the watch pane will show that the first character of the variable string has been replaced with the first character of the constant replacement string.

- Remove the breakpoint
- Right click on the ‘DisplayString();’ function call after the loop and select ‘Go to cursor’.

This shows that the variable was initialised at program start up and can be overwritten with ‘TESTTEST’.

The modified string is also displayed on the LCD

You have now run the tutorial code and used many of the common features of the debugger. We suggest that you review the rest of the tutorial code as many functions have important information on the operation of the code, the compiler directives and comments on when they should or must be used. Please refer to Chapter 7 for more information on the project files.

Chapter 7. Project Files

7.1. Standard Project Files

The RSK tutorials are configured so that it is possible to provide the same tutorial code on multiple RSK products. This allows the evaluation of the different processor cores using equivalent code. To achieve this the following files are common between all device cores / Toolchains.

Each of the tutorial files has expanded comment text describing the function of each code entry. Please refer to the source code for greater detail on the purpose and operation of the compiler specific details.

7.1.1. Initialisation code (Resetprg.c)

This is the entry point of the main tutorial code. Depending upon the compiler used this file may be the actual entry point of the software or may be called during the initial setup of the environment.

```
void Initialise(void)
{
    /* Initialize the Vector Base Register with the base address of exception
       vector table. */
    set_vbr((void *)((char *)0));

    /*  Copy initilized data from ROM to RAM.
        dbsct.c file will be used by this function   */
    _INITSCT();

    /* Set up the hardware  */
    HardwareSetup();

    /* Unmask all interrupts */
    set_cr(SR_Init);

    /* Delay for one CPU cycle. */
    nop();

    /* Call the user function */
    main();
```

Initialisation of the variables used in the C compilers and initialisation of stack pointers are completed in the _INITSCT function for the H8 and SH compilers. The call to ‘HardwareSetup()’ will initialise the device hardware and peripherals ready for the tutorial software.

The call to ‘main()’ will start the main demonstration code

7.1.2. Board initialisation code (hwsetup.c / hwsetup.h)

There are four common stages to the configuration of the microcontroller device. The code to demonstrate this is therefore split into four functions. Each function is written specifically for the device supported. The function calls are shown below.

```
/***********************************************************************
Function Name:  HardwareSetup
Description:    Sets up hardware
Parameters:     none
Return value:   none
***********************************************************************/
void HardwareSetup(void)
{
    ConfigureOperatingFrequency();
    ConfigurePortPins();
    EnablePeripheralModules();
    ConfigureInterrupts();
}
/***********************************************************************
End of function HardwareSetup
***********************************************************************/
```

7.1.3. Main tutorial code (Main.c / main.h)

The main tutorial code is common to all tutorial projects. The display initialisation and string display functions operate on the LCD display module. Check compatibility with a ks0066u controller and pin connection on the schematic before connecting an LCD module not supplied by Renesas.

```
/*****************************************************************************
Function Name : main
Description   : Main function.
        This function calls timer, ADC & LCD initialisation functions. The user
        LEDs flashes until the user presses a switch on the RSK.
Parameters    : none
Return value  : none
*****************************************************************************/
void main(void)
{
    /*  Reset the LCD module. */
    InitialiseDisplay();

    /*  Display Renesas Splash Screen. */
    DisplayString(LCD_LINE1,"Renesas");
    DisplayString(LCD_LINE2,NICKNAME);

    /*  Flash the user LEDs for some time or until a key is pressed. */
    FlashLEDs();

    /*  Flash the user LEDs at a rate set by the user potentiomenter (ADC) using
        interrupts. */
    TimerADC();

    /*  Demonstration of initialised variables.
        Use this function with the debugger. */
    Statics_Test();
```

Chapter 8. Additional Information

For details on how to use High-performance Embedded Workshop (HEW), refer to the HEW manual available on the CD or from the web site.

Further information available for this product can be found on the Renesas website at: www.renesas.com/renesas_starter_kits

General information on Renesas Microcontrollers can be found at the following websites.

Global: http://www.renesas.com/

Regional (English language) sites can be accessed from the Global site, or directly by going to:

Europe: http://www.renesas.eu

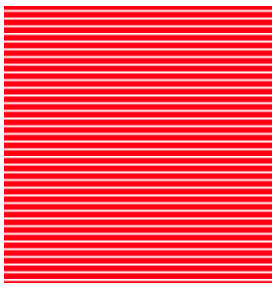
Americas: http://america.renesas.com

Asia: http://sg.renesas.com

Renesas Starter Kit for SH7086
Tutorial Manual
Publication Date Rev.1.00 17.Jan.2008
Published by: Renesas Technology Europe Ltd.

©2008 Renesas Technology Europe and Renesas Solutions Corp., All Rights Reserved.

Renesas Starter Kit for SH7086
Tutorial Manual



Renesas Technology Europe Ltd.
Dukes Meadow, Millboard Road, Bourne End Buckinghamshire SL8 5FH, United Kingdom